 Published in IEEE Transactions on Geoscience and Remote Sensing, Vol. 50, No. 11, pp. 4454-4467.

Best Merge Region-Growing Segmentation With Integrated Nonadjacent Region Object Aggregation

James C. Tilton, *Senior Member, IEEE*, Yuliya Tarabalka, *Member, IEEE*, Paul M. Montesano, and Emanuel Gofman

***Abstract*—Best merge region growing normally produces segmentations with closed connected region objects. Recognizing that spectrally similar objects often appear in spatially separate locations, we present an approach for tightly integrating best merge region growing with nonadjacent region object aggregation, which we call hierarchical segmentation or HSeg. However, the original implementation of nonadjacent region object aggregation in HSeg required excessive computing time even for moderately sized images because of the required inter-comparison of each region with all other regions. This problem was previously addressed by a recursive approximation of HSeg, called RHSeg. In this paper, we introduce a refined implementation of nonadjacent region object aggregation in HSeg that reduces the computational requirements of HSeg without resorting to the recursive approximation. In this refinement, HSeg's region intercomparisons among non-adjacent regions are limited to regions of a dynamically determined minimum size. We show that this refined version of HSeg can process moderately sized images in about the same amount of time as RHSeg incorporating the original HSeg. Nonetheless, RHSeg is still required for processing very large images due to its lower computer memory requirements and amenability to parallel processing. We then note a limitation of RHSeg with the original HSeg for high spatial resolution images and show how incorporating the refined HSeg into RHSeg overcomes this limitation. The quality of the image segmentations produced by the refined HSeg is then compared with other available best merge segmentation approaches. Finally, we comment on the unique nature of the hierarchical segmentations produced by HSeg.**

***Index Terms*—Image analysis, image classification, image region analysis, image segmentation, object detection.**

Manuscript received September 20, 2010; revised January 6, 2012; accepted February 2, 2012. This work was supported in part by the Applied Information Systems Research program of the U. S. National Aeronautics and Space Administration (NASA), by the Internal Research and Development program at NASA's Goddard Space Flight Center, and by the Marie Curie Research Training Network "Hyper-I-Net."

J. C. Tilton is with the Computational and Information Sciences and Technology Office, NASA Goddard Space Flight Center, Greenbelt, MD 20771 USA (e-mail: James.C.Tilton@nasa.gov).

Y. Tarabalka was with the Computational and Information Sciences and Technology Office, NASA Goddard Space Flight Center, Greenbelt, MD 20771 USA. She is currently with the Team AYIN, INRIA, 06902 Sophia Antipolis, France (e-mail: yuliya.tarabalka@inria.fr).

P. M. Montesano is with Sigma Space Corporation, Lanham, MD 20706 USA co-located with the Biospheric Sciences Branch, NASA Goddard Space Flight Center, Greenbelt, MD 20771 USA (e-mail: Paul.M.Montesano@nasa.gov).

E. Gofman is with the IBM Haifa Research Laboratories, Haifa 31905, Israel (e-mail: gofman@il.ibm.com).

Color versions of one or more figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TGRS.2012.2190079

I. Introduction

IMAGE segmentation is the partitioning of an image into related sections or regions. For remotely sensed images of the Earth, an example is a map that divides the image into areas labeled by distinct Earth surface covers such as water, snow, and types of natural vegetation, rock formations, crops and other man created objects. In unsupervised image segmentation, the labeled map may consist of generic labels such as region 1, region 2, etc., which may be converted to meaningful labels by a postsegmentation analysis.

Much of the analysis of remotely sensed imagery is currently performed on a pixel-by-pixel basis. While this analysis approach can be satisfactory for some applications, it is usually not fully effective in extracting the information content from remotely sensed imagery, especially from high spatial resolution imagery [1]. The field of object-based image analysis (OBIA) has arisen in recent years to address the need to move beyond pixel-by-pixel analysis [2]. Image segmentation is the first step for most OBIA approaches, and is a key factor in determining the level of performance for these image analysis approaches.

A popular approach for performing image segmentation is best merge region growing. This approach was first fully described in the archival literature by Beaulieu and Goldberg [3], with similar approaches described earlier in conference proceedings [4]-[7]. Beaulieu and Goldberg's hierarchical step-wise optimization (HSWO) is an iterative form of region growing, in which the iterations consist of finding the most optimal or best segmentation with one region less than the current segmentation.

Many variations on best merge region growing have been described in the literature. As early as 1994, Kurita [8] described an implementation of HSWO that utilized a heap data structure [9] for efficient determination of best merges and a dissimilarity criterion based on minimizing the mean squared error between the region mean image and original image. More recently, a series of papers published by the Leibniz Institute of Ecological and Regional Development (IOER) compared a wide range of image segmentation approaches applicable to remotely sensed imagery analysis [10]-[12]. A number of these approaches were based on best merge region growing, including SEGEN and the segmentation approach contained in the eCognition 2.1 software package.

SEGEN [13] is a relatively pure implementation of best merge region growing, optimized for efficiency in performance, memory utilization, and image segmentation quality.

SEGEN adds a number of (optional) procedures to best merge region growing, among them a low-pass filter to be applied on the first stage of the segmentation and outlier dispatching on the last stage. The latter removes outlier pixels and small segments by imbedding them in neighborhood segments with the smallest dissimilarity. SEGEN also provides several parameters to control the segmentation process. A set of "good in average" control values is suggested in [13].

A form of best merge region-growing segmentation lies at the core of the segmentation approach contained in eCognition 2.1 [14], [15]. However the process for selecting the best merges is much more involved than the relatively straightforward evaluation and comparison of region dissimilarity functions utilized by HSWO and SEGEN. The "multiresolution segmentation" approach of eCognition grows regions with the goal of minimizing image object heterogeneity by accounting for both local image texture and the size of groups of pixels [14]. Smaller objects are merged into larger objects during a local optimization procedure that minimizes object heterogeneity while constrained by a scale parameter limiting object size. A larger scale parameter allows more objects to be fused into larger objects. Object heterogeneity is determined by weighted color and shape parameters [15]. Other segmentation procedures can be combined with the multiresolution approach. For example, spectral difference segmentation merges neighbor objects that fall within a user-defined maximum spectral difference. This procedure can be used to merge spectrally similar objects from the segmentation produced by the multiresolution approach.

In complex scenes, such as remotely sensed images of the Earth, objects with similar spectral signatures (e.g., lakes, agricultural fields, buildings, etc.) appear in spatially separated locations. In such cases, it is useful to aggregate these spectrally similar but spatially disjoint region objects together into groups of regions objects that we call region classes. This aggregation may be performed as a post-processing step. However, best merge region growing, as exemplified by HSWO, may be modified to integrate this aggregation directly into the region-growing process. This is the basis of our hierarchical segmentation (HSeg) algorithm. In contrast to the SEGEN and eCognition segmentation approaches, which seek to improve on HSWO through elaborations on the merge control process, HSeg seeks to improve on HSWO by aggregating spectrally similar spatially disjoint region objects.

Unfortunately, the approach taken for spatially disjoint region object aggregation requires excessive computing time in the original formulation of HSeg. A recursive divide-and-conquer approach, called recursive HSeg (RHSeg), was previously developed to overcome this computational problem. In this paper, we introduce for the first time a refined implementation of nonadjacent region object aggregation in HSeg that reduces the computational requirements of HSeg without resorting to the recursive approximation. The key idea of this refinement is that region object aggregation is limited to region objects containing no less than a dynamically specified minimum number of image pixels.

The HSWO, HSeg and RHSeg algorithms naturally produce a segmentation hierarchy in the form a set of several image segmentations at different levels of detail in which the segmentations at coarser levels of detail can be produced from simple merges of regions at finer levels of detail. This hierarchy may be useful for applications that require different levels of image segmentation details depending on the characteristics of the particular image objects segmented. A unique feature of a segmentation hierarchy that distinguishes it from most other multilevel representations is that the segment or region boundaries are maintained at the full image spatial resolution for all levels of the segmentation hierarchy.

This paper is organized as follows. First, we provide a full description of the original HSeg and RHSeg algorithms (with certain details provided in appendices). We then introduce our refinement of HSeg and note how this refinement of HSeg impacts RHSeg. The computational demands of HSWO, the original HSeg, RHSeg utilizing the original HSeg, the refined HSeg algorithm, and RHSeg utilizing the refined HSeg are compared for a Landsat Thematic Mapper image. Next, we present an approach for evaluation of image segmentation quality and describe three data sets to be utilized in our evaluation. We then show that the refined HSeg algorithm leads to improved flexibility in segmenting moderate- to large-sized high spatial resolution images. We follow this with a comparison of the quality of segmentation-based classification results for the refined version of HSeg with similar classification results from HSWO, SEGEN and Definiens 8.0 (a more recent version of the eCognition 2.1 segmentation approach). The paper concludes with a discussion of the unique nature of the hierarchical set of region class segmentations produced by HSeg or RHSeg.

II. Original HSeg and RHSeg Algorithms

The general ideas behind the original HSeg and RHSeg algorithms were initially described in an early conference proceedings paper [16] and a nearly complete description was first published in [17]. For the first time in open literature, we provide in this section and associated appendices a full complete description of these algorithms.

A. HSeg

The original HSeg algorithm augments best merge region growing with the inclusion of constrained merging of spatially non-adjacent regions, as controlled by the input parameter S_{wght}. This parameter, which can vary from 0.0 to 1.0, controls the relative importance of spatially adjacent and spatially non-adjacent region merges. The analysis flow of HSeg is depicted in Fig. 1, and the algorithm is as follows:

1) Initialize the segmentation by assigning each image pixel a region label. If a presegmentation is provided, label each image pixel according to the presegmentation. Otherwise, label each image pixel as a separate region.

2) Calculate a dissimilarity criterion value d between all pairs of regions. (If S_{wght}.= 0.0, the dissimilarity

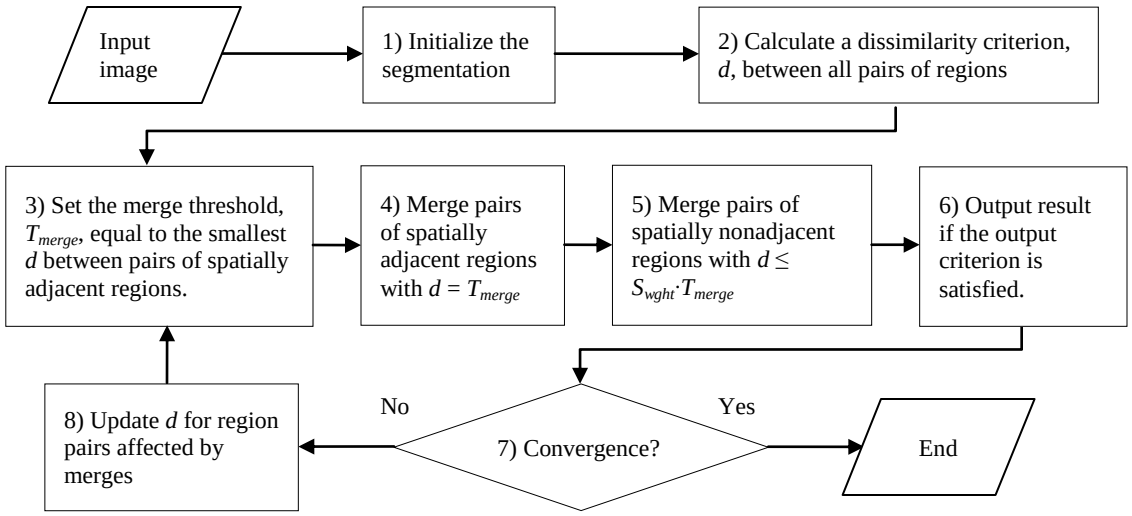


Fig. 1. Analysis flow of the HSeg algorithm.

criterion only needs to be calculated between all pairs of spatially adjacent regions.)

3) Set the merge threshold T_{merge} equal to the smallest dissimilarity criterion value d between pairs of spatially adjacent regions.
4) Merge pairs of spatially adjacent regions with $d = T_{merge}$.
5) If $S_{wght} > 0$, merge pairs of non-adjacent regions with $d \leq S_{wght} \cdot T_{merge}$.
6) Output the segmentation result if the output criterion is satisfied (more on this later).
7) Stop if convergence has been achieved. Otherwise, go to step 8. Convergence is normally considered to be achieved when a specified number of regions has been reached (by default, two regions).
8) Update the dissimilarity criterion values d for the regions affected by merges, and return to step 3.

Since segmentation results with a large number of regions are usually severely oversegmented and thus not of interest, HSeg does not normally output the hierarchical segmentation results until the number of regions reaches a user specified value (by default, 255 regions). After that point, HSeg normally outputs a subsequent hierarchical segmentation result at the iteration just prior to the iteration at which any region would be involved in more than one merge since the last result was output. Alternatively, HSeg can be set to output hierarchical segmentation at a user specified list of number of regions or list of merge thresholds.

One can select from a number of criteria for evaluating how dissimilar one region is from another in HSeg. These dissimilarity criteria include criterion based on vector norms, minimizing the mean square error difference or the change in entropy between the region mean image and the original image, among others (see [18]). We describe in Appendix A the dissimilarity criteria used in tests reported in this paper.

When $S_{wght} = 0.0$, spatially nonadjacent region merges (step 5) are not performed, and HSeg reduces to straightforward best merge region growing. This serves as our implementation of HSWO. With $S_{wght} = 1.0$, merges between spatially adjacent and spatially nonadjacent regions are given equal priority. For values of S_{wght} between 0.0 and 1.0, spatially adjacent merges are given priority over spatially nonadjacent merges by a factor of $1.0/S_{wght}$. Thus for $S_{wght} > 0.0$, region objects (i.e., spatially connected regions) may be aggregated into spatially disjoint groupings that we call region classes.

We noted in the introduction that Kurita [8] showed that a heap data structure [9] can be utilized for an efficient implementation of HSWO. We use a modified heap structure in our implementation of HSeg. The location of each region object is tracked in the heap data structure through a *heap_index* value. The standard heap algorithms were modified to properly update the region object *heap_index* value every time the heap is adjusted. This *heap_index* is used to quickly find the location in the heap data structure of the regions objects whose heap position needs to be updated.

What regions are considered to be spatially adjacent to other regions depends on the definition of a neighborhood relationship. For our purposes, we use the usual *n*-nearest neighbor concept to define spatial adjacency for image pixels, most commonly 4 nearest neighbors (north, south, east, west; referred to as 4 nn) or 8 nearest neighbors (including the diagonal pixels; referred to as 8 nn). Regions adjacent to a region are the union of the region memberships of the neighbors of the pixels on the boundary of that region.

B. RHSeg

The approach taken for implementing nonadjacent region object aggregation in this original version of HSeg requires excessive computing time. This is because the inclusion of spatially nonadjacent region merging requires the intercomparison of each region to every other region. Since HSeg is normally initialized with single pixel regions, this results in a combinatorial explosion of intercomparisons in the initial stage of the algorithm. In contrast, HSWO requires that each image pixel be initially compared only with its neighboring pixels. The RHSeg approximation to HSeg was devised to overcome this computational problem.

RHSeg recursively subdivides the image data into subsections, and then applies HSeg to the subsections of data that are small enough to be processed relatively quickly. However, RHSeg's subdivision and subsequent recombination of the segmentation results can lead to processing window artifacts in which region boundaries are aligned with the processing window boundaries. This is because some region-merging decisions made by RHSeg in one processing window may have been nonoptimal due to the absence of knowledge concerning regions in other processing windows. RHSeg includes a provision to find and split out pixels that may have been inappropriately merged into a particular region at deeper levels of recursion, and to remerge such pixels into a more appropriate region utilizing the global information available at higher levels of recursion.

By nature of its recursive formulation, RHSeg has a straightforward coarse-grained parallel implementation which was described in [17]. We provide a full description of RHSeg in Appendix B, including the processing window artifact elimination step.

III. Refining HSeg by Limiting Nonadjacent Region Object Aggregation to Objects of a Minimum Size

We introduce in this paper a new refinement of HSeg that significantly reduces the computational requirements of the algorithm. The refinement limits the nonadjacent region-merging (region object aggregation) aspect of the HSeg algorithm to merging region objects containing at least a dynamically specified minimum number of image pixels $P_{\min}$. Such region objects are called "large regions," and the number of such regions is designated as N_{large}. The value of $P_{\min}$ is set initially to the smallest value such that $N_{\text{large}} \leq S_{\max}$, where $S_{\max}$ is a user settable program parameter (defaulted to 1024 regions). In contrast, the original version of HSeg included all regions, regardless of size, in the nonadjacent region merging step of the algorithm.

While this simple refinement of HSeg works well in and of itself for many images, we have noted in our tests some cases where the image segmentation quality adversely suffers from strict adherence to the requirement that $N_{\text{large}} \leq S_{\max}$. To address this problem, we introduce the user settable parameter $S_{\min}$ (defaulted to 512 regions), and allow N_{large} to rise above $S_{\max}$ in certain cases where keeping it below would result in $N_{\text{large}} < S_{\min}$. However, N_{large} is allowed to drop below $S_{\min}$ if forcing it to be no less than $S_{\min}$ would result in $N_{\text{large}} > 6 \cdot S_{\max}$. The goal is to find a value for $P_{\min}$ that keeps N_{large} as close as possible to $S_{\max}$, and preferably less than $S_{\max}$. This is all mediated through setting the value of $P_{\min}$.

The algorithm is as follows: The initial iterations of HSeg are treated as a special case because HSeg is normally initiated with each image pixel as a separate region, making it impossible to find a value for $P_{\min}$ such that $2 < N_{\text{large}} \leq 6 \cdot S_{\max}$ (except for very small images). Therefore, initially, all identical spatially adjacent pixels (those with zero dissimilarity value, if any), are merged to form multiple pixel regions. Then, additional iterations of spatially adjacent region merges are performed as necessary until a value for $P_{\min}$ can be found that results in $2 < N_{\text{large}} \leq S_{\max}$. After HSeg is initialized this way, HSeg continues with alternating spatially adjacent and spatially nonadjacent region merges with the value of $P_{\min}$ set as follows: initially set $P_{\min}$ to the smallest value such that $N_{\text{large}} \leq S_{\max}$. If this results in $N_{\text{large}} < S_{\min}$, the value of $P_{\min}$ is reduced by one (unless it is already equal to one) and the value of N_{large} with this new value of $P_{\min}$ is determined. If this new value of $P_{\min}$ results in $N_{\text{large}} > 6 \cdot S_{\max}$, the value of $P_{\min}$ is incremented back up by one. Finally, if this later adjustment results in $N_{\text{large}} < 2$, the value of $P_{\min}$ is again reduced by one, regardless of whether or not this results in $N_{\text{large}} > 6 \cdot S_{\max}$.

For processing efficiency, the value of $P_{\min}$ is not checked for adjustment every iteration. Whenever the value of $P_{\min}$ is changed, "local" values of $S_{\max}$ and $S_{\min}$ are determined (call them $s_{\max}$ and $s_{\min}$), and the value of $P_{\min}$ is checked only when the number of "large regions" becomes less than $s_{\min}$ (and the value of $P_{\min}$ is more than one) or becomes larger than $s_{\max}$. This prevents performing unnecessary computations when it is unlikely that the value of $P_{\min}$ would be changed.

The values of $s_{\min}$ and $s_{\max}$ are recalculated whenever $P_{\min}$ is checked for adjustment. For $s_{\min}$, let $s_{\min} = N_{\text{large}}$. However, if $N_{\text{large}} \leq S_{\max}$, compute $temp = S_{\max} - 2 \cdot (S_{\max} - N_{\text{large}})$, and if $temp > S_{\min}$, let $s_{\min} = temp$. If $s_{\min} > N_r$ (the current number of regions, both "large" and "small"), let $s_{\min} = N_r$. Compute $\max S_{\min} = S_{\max} - 0.05 \cdot (S_{\max} - S_{\min})$. If $s_{\min} > \max S_{\min}$, let $s_{\min} = \max S_{\min}$. For $s_{\max}$, if $N_{\text{large}} > S_{\max}$, let $s_{\max} = N_{\text{large}}$. Otherwise let $s_{\max} = S_{\max}$.

Like the original versions, the refined version of HSeg includes an option for small region merge acceleration (see Appendix A).

IV. Comparison of the Computational Demands of the Original and Refined Versions of HSeg

Timing tests comparing the original and refined versions of HSeg were performed on portions of a six-band Landsat Thematic Mapper data set which was collected on May 28, 1999 from over MD and VA. These tests also include timing tests for RHSeg incorporating either the original or refined version of HSeg. The square root of the Band Sum Mean Squared Error ($\text{BSMSE}^{1/2}$) dissimilarity criterion was utilized with 8 nn neighborhoods and no small region merge acceleration (see Appendix A). The computer used for all tests, except for the parallel processing tests, has an AMD Phenom II 920 2.8-GHz 64-b Quad-Core processor with 8192-MB RAM. The parallel tests were performed on the Discover system at the NASA Center for Climate Simulation (NCCS). The nodes utilized on the Discover system consist of 2 quad-core 2.8 GHz Intel Xeon Nehalem processors.

Table I compares the wall clock run times for the original and refined versions of HSeg. For very small images (such as 64 × 64 pixels), the wall clock run times for the two versions are similar. As the image sizes get larger, the run times of the two versions diverge more and more until, at an image size of

TABLE I
COMPARISON OF WALL CLOCK RUN TIMES FOR THE ORIGINAL AND REFINED VERSIONS OF HSEG. TIMES IN MIN:SEC.

Image Size	Original version of HSeg			Refined version of HSeg		
	S_{wght}					
	0.2	0.5	1.0	0.2	0.5	1.0
64×64	<0:01	0:02	0:09	0:02	0:02	0:01
128×128	0:06	0:39	3:05	0:08	0:09	0:09
256×256	3:38	19:20	71:24	0:18	0:21	0:46
512v512	46:48	363:29	> 1000:00	0:57	1:07	3:05

TABLE II
COMPARISON OF WALL CLOCK RUN TIMES FOR HSWO, THE REFINED VERSION OF HSEG, AND RHSEG INCORPORATING THE REFINED VERSION OF HSEG. TIMES IN MIN:SEC.

Image Size	HSWO	HSeg			RHSeg		
	S_{wght}						
	0.0	0.2	0.5	1.0	0.2	0.5	1.0
512 ×512	0:16	0:57	1:06	2:57	1:18[1]	1:36[1]	3:18[1]
1024 ×1024	1:28	5:23	3:56	7:51	3:37[1]	4:26[1]	8:49[1]
2048 ×2048	14:05	80:37	33:19	32:49	13:40	15:09	24:46
4096 ×4096	*	*	*	*	84:12	67:20	98:15
6912 ×6528	*	*	*	*	255:35	185:02	303:35

[1] With L_r = 2 (default is L_r = 1 for 512×512 and 1024×1024 images).
* Test could not be run due to computer RAM memory limitations.
NOTE: RHSEG was run with the default values for L_r = 2, 3, and 4 for image sizes 2048×2048, 4096×4096 and 6912×6528, respectively.

TABLE III
COMPARISON OF WALL CLOCK RUN TIMES FOR RHSEG INCORPORATING THE ORIGINAL AND REFINED VERSIONS OF HSEG. TIMES IN MIN:SEC.

Image Size	RHSeg with original HSeg			RHSeg with refined HSeg		
	S_{wght}					
	0.2	0.5	1.0	0.2	0.5	1.0
1024×1024	2:12	5:01	11:58	3:37	4:26	8:49
2048×2048	10:40	21:58	50:10	13:40	15:09	24:46

NOTE: RHSeg with the original version of HSeg was run with L_r = 6 and 7 for image sizes 1024×1024 and 2048×2048, respectively. RHSeg with the refined version of HSeg was run with L_r = 2 for both image sizes.

256 × 256 pixels, the refined version of HSeg is from 12 to over 93 times faster than the original version. This difference in run times becomes even more pronounced for larger images. For example, for S_{wght} = 1.0, the refined version of HSeg took less than 3 minutes to complete for a 512 × 512 image, while the original version did not complete even after running for over 16 h.

TABLE IV
COMPARISON OF THE WALL CLOCK RUN TIMES FOR RHSEG INCORPORATING THE REFINED VERSION OF HSEG RUN WITH ONE CPU VERSUS THE SAME CONFIGURATION OF RHSEG RUN WITH MULTIPLE CPUS ON THE NCCS DISCOVER CLUSTER (SEE TEXT). RUN TIMES IN MIN:SEC.

Image Size	RHSeg 16 CPUs for 2048×2048, 64 CPUs for 4096×4096 256 CPUs for 6912×6528				RHSeg 1 CPU			
	L_r	S_{wght}			L_r	S_{wght}		
		0.2	0.5	1.0		0.2	0.5	1.0
2048 ×2048	3 4 5	2:37 1:54 1:41	4:32 4:04 2:54	13:55 13:03 6:26	2 3	13:40 20:02	15:09 24:16	24:46 45:26
4096 ×4096	4 5 6	5:48 3:23 2:17	10:46 7:29 4:11	35:20 23:57 10:32	3 4	84:12 87:57	67:20 124:39	98:15 180:11
6912 ×6528	5 6 7	3:35 3:36 2:08	15:58 7:38 4:10	55:31 26:34 9:15	4 5	255:35 437:54	185:02 515:18	303:35 460:15

Table II compares the wall clock run times for refined version of HSeg and RHSeg incorporating the refined version of HSeg. Also included in this table are run times for HSWO (actually HSeg run with S_{wght}= 0.0) that show the effect of the region aggregation step on run time. These timing results show that, with the new refinement, the wall clock run times for HSeg are now similar to (less than a factor of two different) the run times for RHSeg for image up through 1024 × 1024 in size. Note that, while RHSeg can process images as large as a full Landsat scene (the 6912 × 6528 image size), HSeg (and HSWO) ran out of RAM memory for the tests with images larger than 2048 × 2048. Additionally, the HSeg processing times for the 2048 × 2048 image size was longer than the RHSeg processing times due to page swapping.

Table III compares the wall clock run times for RHSeg incorporating the original version of HSeg and RHSeg incorporating the refined version of HSeg. These results show that the wall clock run times for these two instances of RHSeg are similar.

Table IV compares the wall clock run times for RHSeg incorporating the refined version of HSeg on a parallel cluster versus the same configuration of RHSeg on a single processor. These results show that RHSeg is still a useful approximation of HSeg because its straightforward coarse-grained parallel implementation provides substantial run time improvements. Using 256 CPUs, the parallel version of RHSeg provides an 8 to 122 times speed up versus corresponding (same L_r) 1 CPU RHSeg runs on the full Landsat scene (6912 × 6528). Note that the number of recursive levels (L_r) that produces the smallest processing time differs between the one-CPU and multiple- CPU cases. This is because using a larger number of recursive levels more effectively exploits parallel processing.

V. Evaluation of Image Segmentation Quality

In order to demonstrate the effectiveness of the new versions of HSeg and RHSeg, we now turn to the problem of evaluating image segmentation quality. We have chosen to assess segmentation results using a region-based classification approach [19]. The first step is to perform a pixelwise classification of an image data set. Then, a region classification is obtained by assigning each spatially connected region from the segmentation result to the most frequently occurring class within the region. While this is called the *majority vote* rule in [19], this is a *plurality vote* (PV) rule by a strict definition of the terms. The PV term is used herein.

We have chosen to create our pixelwise classification using the support vector machine (SVM) classifier. This classifier was chosen because it has been shown to perform extremely well in classifying high-dimensional data (such as hyperspectral data) with a limited number of training samples [20]. The particular SVM classifier utilized is the multiclass pairwise (one versus one) SVM classifier, with the Gaussian radial basis function (RBF) kernel, by means of the LIBSVM library [21].

We now describe the hyperspectral data sets used in our tests, and how we utilize the associated reference data for training and testing. The spectral angle mapper (SAM) criterion was used in HSeg, since this criterion is widely accepted for hyperspectral analysis (see Appendix B). We also used 4 nn neighborhoods in our tests since we found that our results were better with 4 nn than with 8 nn neighborhoods.

A. *Washington DC Mall HYDICE Hyperspectral Data Set*

The Washington DC Mall hyperspectral data set was obtained through Purdue University's MultiSpec freeware project [22]. This data set was acquired with the HYDICE sensor, which collected data in the 0.4-2.4-μm region of the visible and infrared spectrum. The data set analyzed contains 191 spectral bands with spatial dimensions of 307 columns and 1208 rows and ground spatial resolution of 1.5 m/pixel. The ground reference data consist of seven classes of interest: roofs, street, graveled path, grass, trees, water and shadow. A three-band false color image and the reference data are shown in Fig. 2.

Two disjoint training data sets were created by randomly selecting pixels from the complete ground reference data. The SVM training set was used to train the SVM classifier, and the segmentation training set was used for segmentation algorithm parameter optimization (as discussed in a following section). Table V lists the number of pixels in the training and test sets for each ground cover class. The SVM training set was selected with the somewhat arbitrary goal of selecting about 200 pixels for each class (with more for the classes with larger numbers of reference pixels), and roughly the same number for the segmentation algorithm training set. Since this training set selection process worked well with this and the other hyperspectral data sets, it was not further optimized as it is not a focus of this study.

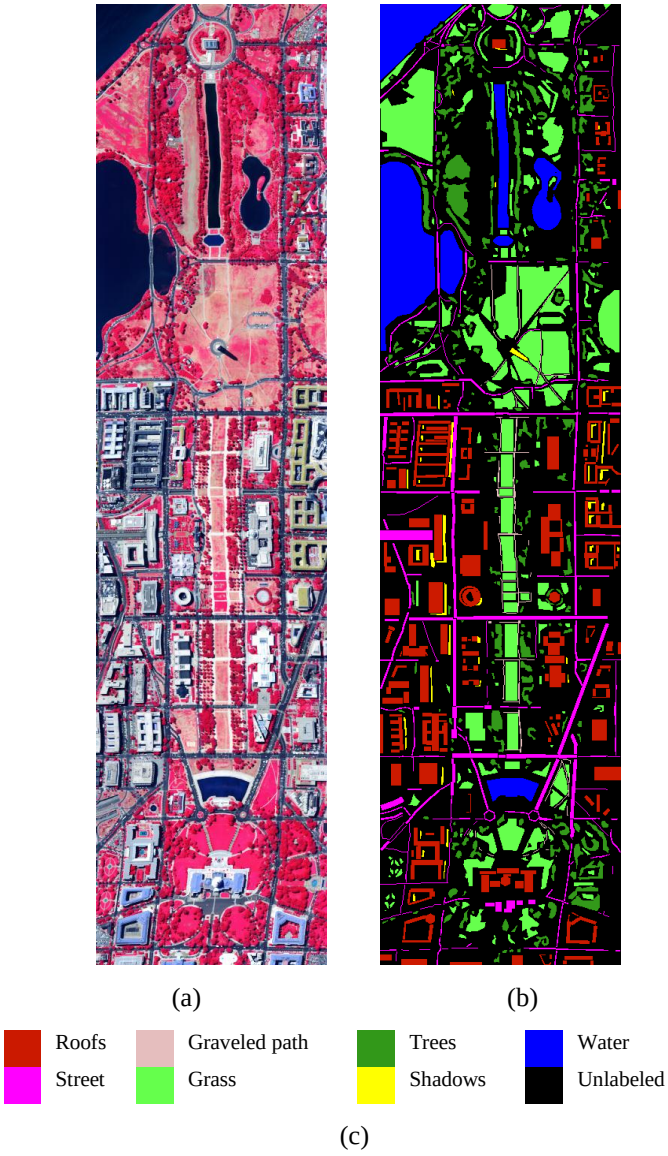


Fig. 2. (a) Three-band false color image of the Washington DC Mall hyperspectral data set (RGB = bands 60, 27 and 17). (b) Reference data. (c) Color key. The training data for the SVM classifier and segmentation optimization are randomly selected pixels from the reference data.

B. *University of Pavia ROSIS Data Set*

The University of Pavia data set was recorded by the Reflective Optics System Imaging Spectrometer (ROSIS) over the University of Pavia, Pavia, Italy. The image is 610 × 340 pixels in size, with a spatial resolution of 1.3 m. The ROSIS sensor has 115 spectral channels, with a spectral range of 0.43-0.86 μm. The 12 noisiest channels were removed, and the remaining 103 spectral bands were used in this experiment. See [19] and [23] for more details on this data set. The reference data contain nine ground cover classes: asphalt, meadows, gravel, trees, metal sheets, bare soil, bitumen, bricks and shadows. A three-band false color image of this data set and the ground reference data are shown in Fig 3.

TABLE V
NUMBER OF PIXELS IN THE SVM TRAINING, SEGMENTATION TRAINING AND TEST SETS FOR THE WASHINGTON DC MALL DATA SET.

	Roofs	Street	Graveled Path	Grass	Trees	Water	Shadow	Total
SVM Training	297	176	212	398	197	260	211	1751
Segmentation Training	317	216	212	474	209	275	234	1937
Segmentation Testing	28361	18783	1603	42527	18000	24956	1715	135945
Ground Reference	28975	19175	2027	43399	18406	25491	2160	139633

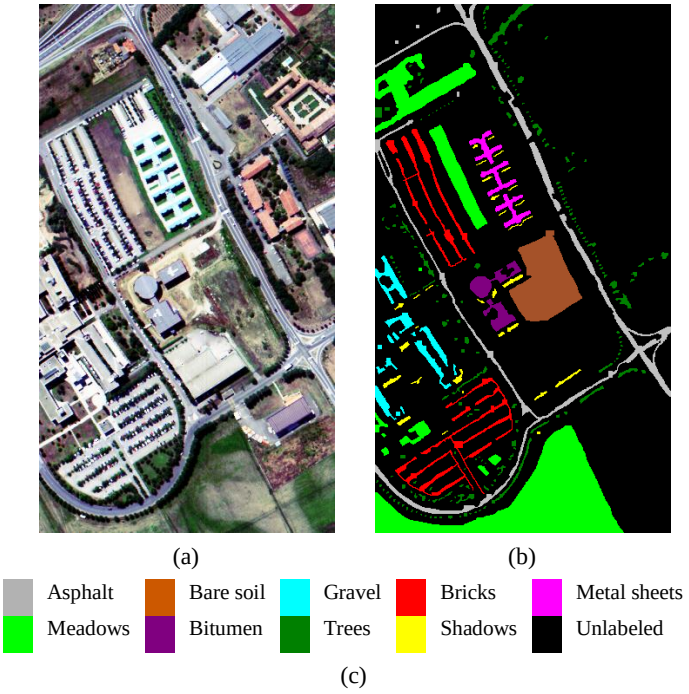


Fig. 3. (a) Three-band false color image of the University of Pavia hyperspectral data set (RGB = bands 56, 33 and 13). (b) Reference data. (c) Color key. The training data for the SVM classifier and segmentation optimization are randomly selected pixels from the reference data.

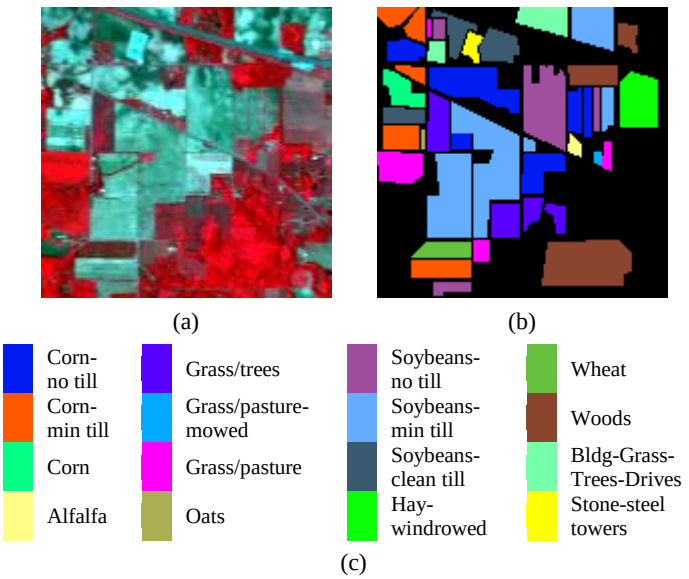


Fig. 4. (a) Three-band false color image of the Indian Pines hyperspectral data set (RGB = bands 47, 24 and 14). (b) Reference data. (c) Color key (black designates unlabeled). The training data for the SVM classifier and segmentation optimization are randomly selected pixels from the reference data.

As with the Washington DC Mall HYDICE data set, SVM and segmentation training sets were created by randomly selecting pixels from the complete ground reference data. The SVM training set was selected with the somewhat arbitrary goal of selecting about 100 pixels for each class (with more for the very large Meadows class), and about double that number for the segmentation training set.

C. Indian Pines AVIRIS Hyperspectral Data Set.

The Indian Pines hyperspectral data set was obtained through Purdue University's MultiSpec freeware project [22]. This data set was acquired by the AVIRIS sensor over the vegetated Indian Pines site in Northwestern Indiana. The image has spatial dimensions of 145 by 145 pixels, with a ground spatial resolution of 20 m/pixel. Twenty water absorption bands (104-108, 150-163, and 220) were removed (per [24]), resulting in a 200-band image that was used in our experiments. The ground reference data contain 16 classes of interest, which represent mostly different types of crops. A three-band false color image and the reference data are shown in Fig. 4.

As with the previous two data sets, SVM and segmentation training sets were created by randomly selecting pixels from the reference data. The number of pixels available for the training set was limited by the small number of reference pixels available for some classes. For classes containing less than 100 reference pixels, roughly 1/3 each were selected for SVM training, segmentation algorithm training and testing. For the other classes, roughly 50 pixels were selected from each class for SVM training and segmentation training and the remainder were utilized for testing (for two classes with over 1000 reference pixels, roughly 100 pixels were selected for the training sets).

For all three data sets, the optimal parameters for the SVM classifier were obtained by fivefold cross validation on the respective SVM training sets. For the University of Pavia data set, the optimal values for C and γ were found to be 32 and 0.5, respectively. $C = 512$ and $\gamma = 2^{-9}$ were found to be optimal for the Indian Pines data set and $C = 32768$ and $\gamma = 2^{-11}$ were found to be optimal for the Washington DC Mall data set. The parameter C controls the amount of penalty during the SVM optimization [25] and parameter γ controls the spread of the Gaussian RBF kernel.

The PV classifications for all segmentation approaches were evaluated in terms of Overall Accuracy (OA), Average Accuracy (AA) and the kappa coefficient (κ). OA is the percentage of correctly classified pixels, AA is the mean of the class-specific accuracies, and κ is the percentage of agreement

(correctly classified pixels) corrected by the number of agreements that would be expected purely by chance [26].

VI. Implications of the HSeg Refinement on Image Segmentation Quality

Subdivision of the image data down to subsections in the range of 1000-4000 pixels was necessary for the original version of HSeg, whereas subdivision down to sub-sections in the range of 250 000-1 000 000 pixels (e.g., image subsections up to about 1024 × 1024 pixels) is all that is required for the refined version of HSeg. We show here that this difference in size of the smallest subsections processed in the RHSeg approximation leads to improved flexibility in performing image segmentations.

High spatial resolution images, such as the Washington DC Mall HYDICE data set, contain many small region objects, which may, or may not, be significant, depending on the goals of the analysis. If these small regions objects are not significant, small region merge acceleration (favored merging of regions smaller than Π_{min} pixels, see Appendix A) may be used in either the original and refined versions of HSeg or in RHSeg incorporating either the original or refined version of HSeg without adversely affecting the results. This can be seen in the results reported in Table VI comparing the plurality vote classification accuracies obtained for RHSeg incorporating the original version of HSeg to those obtained by the refined version of HSeg.

However, if small region objects are significant for the particular analysis application, small region merge acceleration should not be utilized. Table VII shows the results obtained for RHSeg incorporating the original version of HSeg and the results obtained for the refined version of HSeg without small region merge acceleration. While the results for the refined version of HSeg were not adversely affected, not using small region merge acceleration had disastrous effects on the results obtained with RHSeg incorporating the original version of HSeg. This was because processing efficiency required that the initial processing windows contain only 1600 pixels, forcing RHSeg to return results for no more than 1600 region classes at the top level of recursion. The implication of this is one has much improved flexibility in using RHSeg incorporating the refined version of HSeg to segment large images than one had using RHSeg when it incorporated the original version.

VII. Comparison of the Effectiveness of Refined HSeg to HSWO and other Image Segmentation Approaches

Presented here is a comparison of the effectiveness of the refined version of HSeg to HSWO, SEGEN and the segmentation approach from Definiens 8.0 in segmenting hyperspectral data sets. Here, we compare the effectiveness of the segmentation approaches by comparing the classification accuracies obtained by the region-based PV classification approach described earlier.

A. Parameter Optimization

The segmentation training sets were used to optimize parameters for the SEGEN, Definiens 8.0, HSWO, and HSeg

TABLE VI

With Small Region Merge Acceleration. A Comparison of Classification Accuracies on the Washington DC Mall Hyperspectral Data Set for RHSeg Incorporating the Original Version of HSeg and for the Refined Version of HSeg With the Region PV Method and Various Values of S_{WGHT}. Classification Accuracies in Percentage in Terms of OA, AA, and Kappa Coefficient (κ).

	RHSeg incorporating the original HSeg* + PV					Refined version of HSeg + PV				
S_{wght}	0.1	0.2	0.3	0.5	1.0	0.1	0.2	0.3	0.5	1.0
# of region objects	8721	25012	46835	73509	184082	120004	102905	39125	43685	44522
OA	96.53	96.92	96.86	96.42	96.13	96.52	96.58	96.72	96.84	96.89
AA	93.76	95.22	96.21	95.73	95.81	95.61	95.49	94.99	95.75	96.01
κ	95.59	96.10	96.02	95.48	95.11	95.60	95.66	95.85	96.00	96.07

* Π_{min} matched to the maximum value of P_{min} observed for HSeg: For S_{wght} = 0.1, Π_{min} = 72, for S_{wght} = 0.2, Π_{min} = 67, for S_{wght} = 0.3, Π_{min} = 54, for S_{wght} = 0.5, Π_{min} = 36, for S_{wght} = 1.0, Π_{min} = 17.

TABLE VII

Without Small Region Merge Acceleration. A Comparison of Classification Accuracies on the Washington DC Mall Hyperspectral Data Set for RHSeg Incorporating the Original Version of HSeg, and the Refined Version of HSeg With the Region PV Method and Various Values of S_{WGHT}. Classification Accuracies in Percentage in terms of OA, AA, and Kappa Coefficient (κ).

	RHSeg incorporating the original HSeg* + PV					Refined version of HSeg + PV				
S_{wght}	0.1	0.2	0.3	0.5	1.0	0.1	0.2	0.3	0.5	1.0
# of region objects	3525	5685	1970	21442	111445	61369	38106	44080	55282	60267
OA	74.10	85.18	77.95	82.15	96.43	96.95	96.88	96.58	96.65	96.72
AA	66.37	74.93	58.57	74.29	95.87	96.17	96.06	95.86	95.57	95.64
κ	67.34	81.11	71.01	77.46	95.48	96.14	96.05	95.67	95.76	95.86

* Π_{min} = 0.

segmentation algorithms. For SEGEN, the set of control parameters was varied to find the highest kappa coefficient (κ) value (found to be highly correlated with overall classification accuracy). The dispatch outliers stage of SEGEN was found to improve the results for all three data sets, but the low-pass filter was found to be helpful only on the University of Pavia data set. The Definiens 8.0 segmentation parameters for the multiresolution approach were optimized with the goal of creating image objects detailed enough to resolve the features identified in the pixel-based classification while not oversegmenting. The scale parameter and the relative influence of shape/color were systematically adjusted to derive a segmentation that produced an accurate classification at a scale that resolved the features classified in the pixel-level classification. In addition, the spectral difference segmentation algorithm was used to merge spectrally similar objects from the multiresolution segmentation result. The level of segmentation detail was optimized for HSWO and HSeg by selecting the hierarchical level that gave the highest classification accuracy on the segmentation training set. The S_{wght} parameter was also similarly optimized for HSeg, the PV classification was performed over the region object map instead of the region class map, and we checked whether not small region merge acceleration improved the segmentation results. We used 4 nn connectivity for all segmentation approaches since we found that this generally produced the best PV classification accuracy results.

B. PV Classification Results

Tables VIII-X compare the results on each data set of our analysis for the pixelwise SVM classifier and PV over the regions produced by the various segmentation approaches. The segmentation approaches generally produced some improvement over the pixelwise SVM classifier.

The SEGEN+PV case produced the best results for the Washington DC Mall data set, with HSWO+PV and HSeg+PV (without small region merge acceleration) virtually tied for second. Note that since a very small weight (S_{wght} = 0.1) was given to non-adjacent region merges in the HSeg segmentation, the HSeg+PV and HSWO+PV classification results came out very similar.

The HSeg+PV (with small region merge acceleration) case produced the best results for the University of Pavia data set, with the SEGEN+PV case a close second. Note that the HSWO+PV result has much lower classification accuracies than the HSeg+PV result and that a higher weight (S_{wght} = 0.3) was given to nonadjacent region merges in the HSeg segmentation.

The SEGEN+PV case produced the best results for the Indian Pines data sets, with the D8+PV and HSeg+PV cases at a close tie for second. Note that even though a very small weight (S_{wght} = 0.1) was given to nonadjacent region merges in the HSeg segmentation, the HSeg+PV classification accuracies are much higher than the HSWO+PV accuracies.

Figs. 5-7 show the classification maps produced for selected cases. Fig. 5 shows the SEGEN+PV and HSeg+PV classification maps for the Washington DC Mall data set. These two classification maps look very similar. Fig. 6 shows the SEGEN+PV and HSeg+PV classification maps for the University of Pavia data set. The SEGEN+PV classification map is somewhat smoother than the HSeg+PV classification. Fig. 7 shows the SEGEN+PV and HSeg+PV classification

TABLE VIII

COMPARISON OF CLASSIFICATION ACCURACIES ON THE WASHINGTON DC MALL HYPERSPECTRAL DATA SET FOR PER-PIXEL SVM AND WITH THE REGION PV METHOD FOR DEFINIENS 8.0 (D8), SEGEN, HSWO, AND THE REFINED VERSION OF HSEG. HSEG WAS PERFORMED WITHOUT SMALL REGION MERGE ACCELERATION. CLASSIFICATION ACCURACIES IN PERCENTAGE IN TERMS OF OA, AA, KAPPA COEFFICIENT (κ) AND CLASS-SPECIFIC ACCURACIES. HIGHEST VALUES AND VALUES WITHIN 0.15% OF HIGHEST VALUE ARE **BOLDED** IN EACH CATEGORY.

	SVM	D8+PV	SEGEN + PV	HSWO+PV	HSeg+PV S_{wght} = 0.1
# of region objects	NA	22313	23731	53146	61369
OA	95.76	96.87	**97.13**	96.99	96.95
AA	95.54	95.50	95.31	**96.02**	**96.17**
κ	94.64	96.04	**96.37**	96.19	96.14
Roofs	91.30	93.80	**94.00**	**93.94**	**93.90**
Street	94.48	95.57	**96.22**	95.45	95.38
Graveled Path	**94.82**	89.58	88.65	91.83	93.20
Grass	97.49	98.24	**98.66**	98.43	98.30
Trees	96.45	**96.78**	**96.71**	**96.78**	**96.79**
Water	98.39	99.70	**99.91**	99.73	99.72
Shadow	95.86	94.81	93.06	**95.98**	95.86
# bolds	1	1	7	4	3

TABLE IX

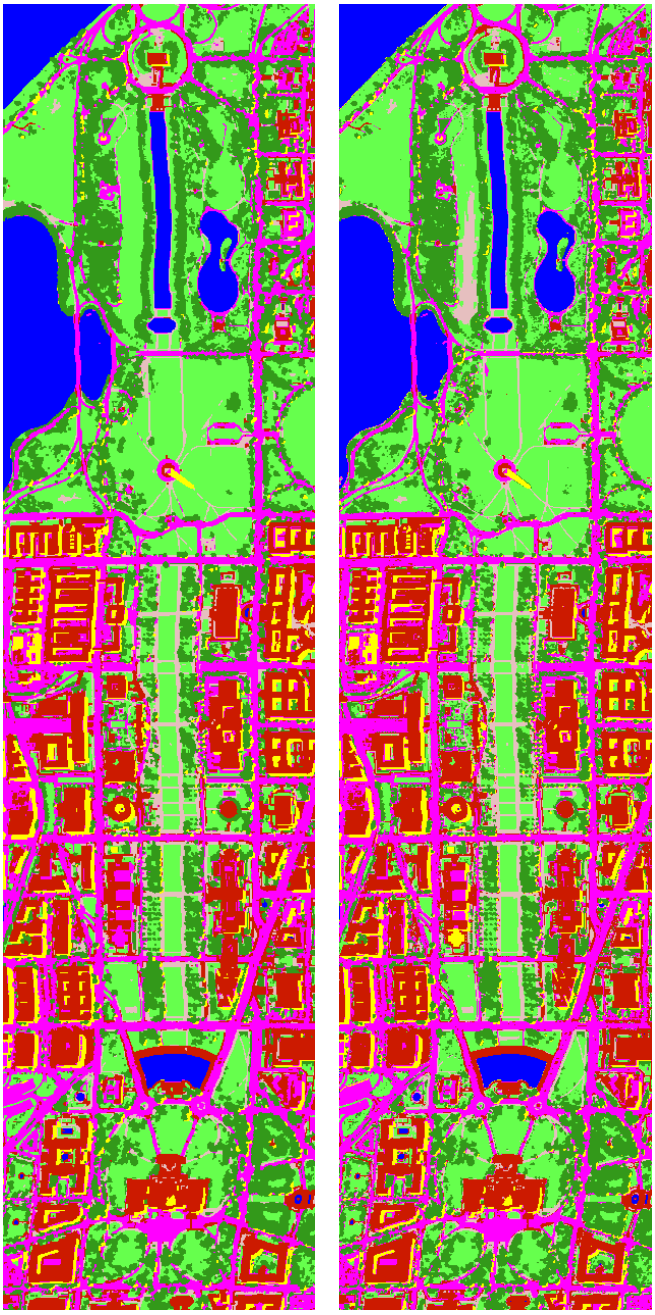
COMPARISON OF CLASSIFICATION ACCURACIES ON THE UNIVERSITY OF PAVIA HYPERSPECTRAL DATA SET FOR PER-PIXEL SVM AND WITH THE REGION PV METHOD FOR DEFINIENS 8.0 (D8), SEGEN, HSWO, AND THE REFINED VERSION OF HSEG. HSEG WAS PERFORMED WITH SMALL REGION MERGE ACCELERATION. PERCENTAGE CLASSIFICATION ACCURACIES IN TERMS OF OA, AA, AND KAPPA COEFFICIENT (κ).

	SVM	D8+PV	SEGEN +PV	HSWO +PV	HSeg+PV S_{wght} = 0.3
# of region objects	NA	4501	4019	74279	30021
OA	89.03	97.54	98.09	95.38	**98.35**
AA	89.56	97.26	97.95	95.50	**98.15**
κ	85.46	96.71	97.45	93.83	**97.79**

TABLE X

COMPARISON OF CLASSIFICATION ACCURACIES ON THE INDIAN PINES HYPERSPECTRAL DATA SET FOR PER-PIXEL SVM AND WITH THE REGION PV METHOD FOR DEFINIENS 8.0 (D8), SEGEN, HSWO, AND THE REFINED VERSION OF HSEG. HSEG WAS PERFORMED WITHOUT SMALL REGION MERGE ACCELERATION. CLASSIFICATION ACCURACIES IN PERCENTAGE IN TERMS OF OA, AA, AND KAPPA COEFFICIENT (κ).

	SVM	D8+PV	SEGEN+ PV	HSWO+PV	HSeg+PV S_{wght} = 0.1
# of region objects	NA	475	1074	5323	4057
OA	76.41	85.27	**87.47**	85.33	86.89
AA	80.77	91.57	**92.31**	86.31	89.83
κ	72.92	82.92	**85.51**	83.07	84.84

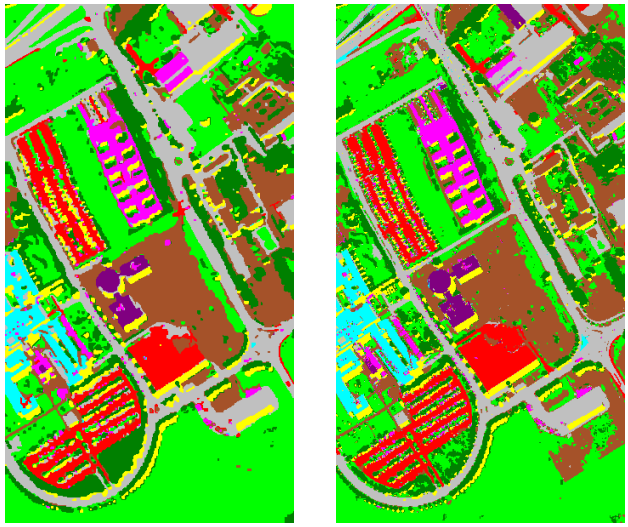


(a) (b)

Fig. 5. Classification maps for the Washington DC Mall hyperspectral data set. (a) PV of the SVM classification over the SEGEN segmentation. (b) PV of the SVM classification over the HSeg segmentation (refined version) with S_{wght} = 0.1 and no small region merge acceleration. The color key is as in Fig. 2.

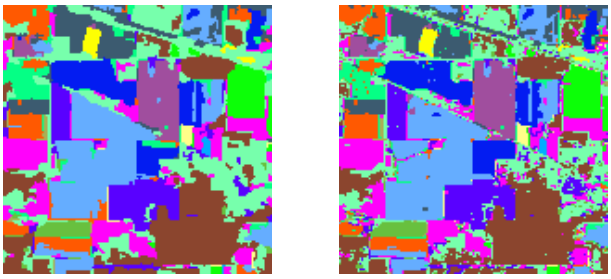
maps for the Indian Pines data set. The SEGEN+PV classification has fewer small regions, due to its built-in methods for reducing the number of small regions.

The classification results presented show that HSeg's integration of nonadjacent region object aggregation in the best merge region-growing process can often improve the segmentation results over those produced by the HSWO approach which limits the region growing process to spatially



(a) (b)

Fig. 6. Classification maps for the University of Pavia data set. (a) PV of the SVM classification over the SEGEN segmentation. (b) PV of the SVM classification over the HSeg segmentation (refined version) with S_{wght} = 0.3 and with small region merge acceleration. The color key is as in Fig. 3.



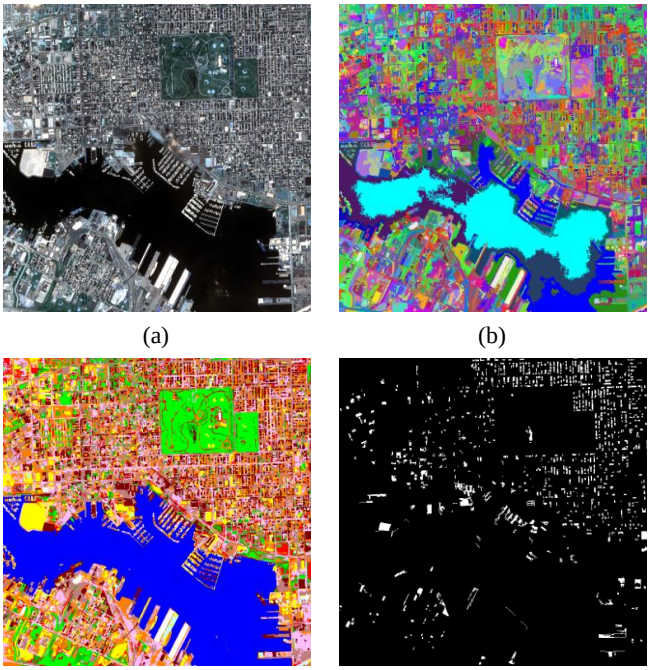
(a) (b)

Fig. 7. Classification maps for the Indian Pines hyperspectral data set. (a) PV of the SVM classification over the SEGEN segmentation. (b) PV of the SVM classification over the HSeg segmentation (refined version) with S_{wght} = 0.1 and without small region merge acceleration. The color key is as in Fig. 4.

adjacent regions. On the other hand, these results do not demonstrate superiority in the comparison between HSeg, Definiens 8.0 and SEGEN. However, these results do show that HSeg is very competitive with these other image segmentation approaches for this type of application. It is interesting to see that simply adding nonadjacent region object merging to the best merge region-growing process improves the segmentation result to the point that the results are competitive with approaches like SEGEN and Definiens 8.0 with their more elaborate merge control processes.

VIII. The Unique Nature of the Hierarchical Segmentatons produced by HSeg and RHSeg

The key unique aspect of HSeg and RHSeg is the tight intertwining of region-growing segmentation, which produces spatially connected region objects, with nonadjacent region object aggregation, which groups sets of region objects



(a) (b) (c) (d)

Fig. 8. (a) True color rendition of a 768×768 pixel portion of an Ikonos image from over Baltimore, MD. (b) Colorization of the HSWO segmentation result at region mean dissimilarity 0.45 (3600 region objects). (c) Colorization of the HSeg segmentation at region mean dissimilarity 0.45 (7,521 region objects grouped into 11 region classes). (d) Region class "10" highlighted from the HSeg result at region mean dissimilarity 0.41.

together into region classes. No other practical operational image segmentation approach has this tight integration of region-growing object finding with nonadjacent region aggregation. The advantage of this tight intertwining is demonstrated in the comparison image segmentation results from the refined HSeg versus HSWO. Figs. 8(a)-(c) shows a true color rendition of a 768 × 768 portion of an Ikonos image from over Baltimore, MD, a colorization of the region object label map from a HSWO result, and a colorization of the region class label map from a corresponding HSeg result. HSWO and HSeg were both run with the BSMSE^{½} dissimilarity criterion until a region mean global dissimilarity of 0.45 was reached. HSeg used S_{wght} = 0.25.

At this level of segmentation detail, a display of the region mean image for either the HSWO or HSeg result (not shown) looks very similar to the original image. However, the HSeg segmentation provides a much more compact description of the data through just 11 region classes. The HSWO segmentation requires 3600 region objects to represent the data. HSeg's tight integration of region-growing object finding and nonadjacent region aggregation enables a compact but high quality representation of the image information content. Also, the natures of the underlying image segmentations are very different. For example, the waters of the Baltimore Inner Harbor are fragmented into several region objects in the HSWO result while these same waters of consolidated into just one region class in the HSeg result. The HSeg result also aggregates spatially disjoint but spectrally similar region objects into single region classes, allowing the identification of spatially disjoint areas of similar ground cover, such as evident for the green colored region class in Fig. 8(c) that corresponds to vegetation. No such direct association is possible in the HSWO result shown in Fig. 8(b). Other similar region class associations are also evident in an inspection of the HSeg result of 8(c), such as building roofs and the road network.

HSeg's unique capability of automatically grouping of spatially connected region objects into region classes provides additional advantages in utilizing HSeg segmentations in OBIA applications. Consider the patterning of dark roofs evident throughout Fig. 8(a). These roofs are labeled as region class "10" in the HSeg segmentation result at region mean dissimilarity 0.41, which is highlighted in white in Fig. 8(d). Note a certain regularity of the roof pattern to the southeast, east and north of Patterson Park. This area is generally an older residential area, with a few businesses interspersed. The roof pattern to the southwest and west of Patterson Park appears somewhat different. This area has a denser concentration of businesses and apartment complexes. Pixel-based analysis could never detect this difference in spatial patterning whereas detection of such spatial patterning should be possible with an OBIA approach. The assumption made here is that if the spatial pattern detection system built into the human eye-brain system can detect it, a sufficiently sophisticated OBIA approach should also be able to detect it. It is the capability of HSeg to find region objects and group nonadjacent region objects into region classes that makes patterns like these roof patterns accessible to OBIA. Preliminary work investigating utilizing a graph-based knowledge discovery system to identify such patterns was reported in [27] and [28]. HSeg could also be utilized as an input to an approach for classifying patterns of land cover described recently in [29].

IX. Summary and Discussion

This paper has presented HSeg as a form of best merge region-growing segmentation that tightly integrates nonadjacent region object aggregation with the usual region object growing. Presented also is RHSeg, the recursive approximation of HSeg, as an approach for reducing the computational requirements of nonadjacent region object aggregation in HSeg. RHSeg recursively subdivides the image data into subsections that can be efficiently processed and subsequently recombines the segmentation results from the subsections. Since this recombination of segmentation results from the subsections can sometimes lead to processing window artifacts, RHSeg includes a processing window artifact elimination step, which is fully described in this paper for the first time in the open literature.

This paper has also introduced a new refined version of HSeg which reduces the computational requirements of the original version of HSeg by limiting the region object aggregation step to regions containing a dynamically varied minimum number of pixels. We show that the refined version

of HSeg produces similar segmentation results in similar computation times to those produced by RHSeg incorporating the original version of HSeg. However, RHSeg utilizing the refined version of HSeg is still needed to process large images due to its lower needs for computer memory and the availability of a straightforward parallel implementation. We also showed that incorporating the refined version of HSeg into the RHSeg processing scheme improved the flexibility of RHSeg in performing segmentation of large images.

We then compared the effectiveness of HSWO and HSeg to two other image segmentation approaches using a PV region-based classification approach on three quite different hyperspectral image data sets, fed by a pixelwise classification using the SVM classifier. We found the refined version of HSeg to be competitive with other image segmentation approaches (Definiens 8.0 and SEGEN) for this type of application and often superior to the HSWO approach.

Finally, we have noted the unique nature of the HSeg hierarchical segmentations and have examined the potential advantages of utilizing HSeg in OBIA. HSeg's automatic grouping of spatially connected region objects into region classes provides unique advantages for OBIA applications. This grouping leads to high spatial fidelity in the image segmentation results and directly leads to opportunities for developing analysis approaches for detecting spatial patterning.

We encourage others to experiment with HSeg and RHSeg in their image analysis applications. Examples of recent earth science projects funded by NASA that utilize HSeg and RHSeg are reported on in [30] and [31]. A full featured demonstration version of the latest version of HSeg/RHSeg may be obtained directly from the corresponding author or through the web site:

http://ipp.gsfc.nasa.gov/ft_tech_rhseg.shtm.

Certain aspects of this software are subject to patent Nos. US 6,895,115 B2 and 7,697,759 issued by the United States Patent and Trademark Office.

Appendix A
Dissimilarity Criterion

Our implementation of HSWO, HSeg and RHSeg offers a number of criteria for evaluating the dissimilarity of one region versus another. For a complete list see [18]. We briefly describe here the two dissimilarity criteria used in tests reported in this paper.

One dissimilarity criterion is based on minimizing the increase of mean squared error between the region mean image and the original image data. The BSMSE between regions X_i and X_j with region mean vectors, u_i and u_j, and region size (number of pixels) n_i and n_j is given by

$$d_{BSMSE}(X_i, X_j) = \frac{n_i n_j}{(n_i + n_j)} \sum_{b=1}^{B} (\mu_{ib} - \mu_{jb})^2 \quad \text{(A-1)}$$

where $u_i = (\mu_{i1}, \mu_{i2}, \ldots, \mu_{iB})^T$ (similarly for u_j). To keep the dissimilarity criteria dimensionality consistent with other criterion utilized by HSWO, HSeg and RHSeg, the square root of this criterion (BSMSE^{½}) is used [18].

The second dissimilarity criterion selected for use in this study is the SAM criterion, which is widely used in hyperspectral image analysis [32]. This criterion determines the spectral similarity between two spectral vectors by calculating the "angle" between the two spectral vectors. The spectral angle θ between the region mean vectors, u_i and u_j, of regions X_i and X_j is given by:

$$\theta(u_i, u_j) = \arccos\left(\frac{u_i \circ u_j}{\|u_i\|_2 \|u_j\|_2}\right) = \arccos\left(\frac{\sum_{b=1}^{B} \mu_{ib}\mu_{jb}}{\left(\sum_{b=1}^{B} \mu_{ib}^2\right)^{1/2}\left(\sum_{b=1}^{B} \mu_{jb}^2\right)^{1/2}}\right), \quad \text{(A-2)}$$

Note that the value of the SAM criteria, $\theta(u_i,u_j)$, ranges from 0.0 for the most similar vectors up to $\pi/2$ for the most dissimilar vectors.

We have found that segmentations produced using the SAM criterion and certain other criterion, such as those based on vector norms, tend to contain many small regions (see [33]). These small regions can cause difficulties in HSeg because of the need to consider them in the region aggregation step (as discussed in Section VI). Because of this, a bias factor was introduced to encourage or accelerate the merging of small regions into larger ones. Let Π_{min} be a user settable parameter used in calculating this merge acceleration factor, M_A, which is multiplied times the dissimilarity criteria. For two regions of size (number of pixels) n_1 and n_2, let $\Pi_i = \min(n_i, \Pi_{min})$ for i = 1,2. Then

$$M_A = \frac{\left(\frac{\Pi_1 * \Pi_2}{(\Pi_1 + \Pi_2)}\right)^{1/2}}{\left(\frac{\Pi_{min} * \Pi_{min}}{(\Pi_{min} + \Pi_{min})}\right)^{1/2}} = \left(\frac{2 * \Pi_1 * \Pi_2}{\Pi_{min} * (\Pi_1 + \Pi_2)}\right)^{1/2}. \quad \text{(A-3)}$$

Note that, if both n_1 and $n_2 \geq \Pi_{min}$, M_A = 1.0. This factor accelerates the merging of regions with size less than Π_{min} into larger regions. For the SAM and several other criteria, the default value of Π_{min} is set at 200 for the original versions of HSeg and RHSeg. However, since the mean squared error and entropy based criterion have a natural bias against small regions, the default value for Π_{min} is set at 0 (i.e., making M_A always = 1.0) for those criterion.

As noted earlier, the refined version of HSeg also includes an option for small region merge acceleration. In this case when this option is selected, the merge acceleration factor, M_A, is employed when one of the regions has size less than P_{min}. However, instead of setting $\Pi_{min} = P_{min}$, Π_{min} is set equal to the maximum of Π_1 and Π_2 in eq. (A-3). This modification reduces the dissimilarity criteria updating necessary when the value P_{min} changes.

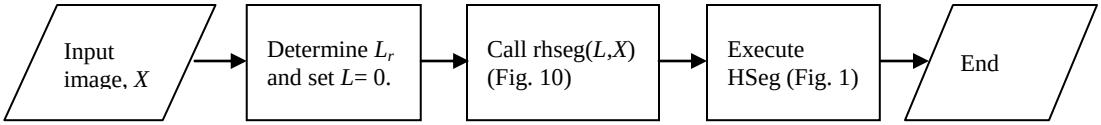


Fig. 9. Analysis flow of the RHSeg algorithm. L_r is determined as the number times the input image must be subdivided to achieve a small enough image size for efficient processing with HSeg.

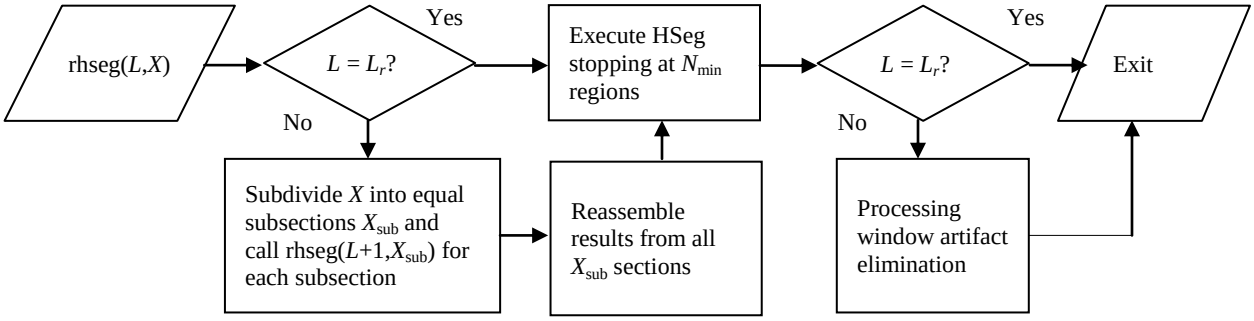


Fig. 10. The analysis flow of the recursive function rhseg(L,X). N_{min} is equal to ¼ the number of pixels in the subimage processed at the deepest level of recursion.

APPENDIX B
THE RHSEG ALGORITHM

This appendix provides a description of RHSeg algorithm, including a full description of the processing window artifact elimination step that has not been published previously in the open literature. The analysis flow for RHSeg, which has been implemented for both 2- and 3-D data [17], is shown in Figs. 9 and 10.

RHSeg recursively subdivides the image data X into smaller subsections, and then applies HSeg to the smaller subsections of data. The number of times the image data is subdivided L_r depends on the size of the image data and the maximum desired size for the image subsection processed at the deepest level of recursion. For RHSeg utilizing the refined version of HSeg, the default for this subimage size is set to 1 048 576 (=1024 × 1024) pixels. In addition, the number of subsections the image is divided into at each recursive level is adjusted to achieve near equal image dimensions at the deepest level of recursion. For example, for a 2-D image with 2048 columns and 4096 rows, the image is first subdivided into two subsections of 2048 columns and 2048 rows and then further subdivided into four subsections of 1024 columns and 1024 rows. In this case L_r = 2.

As noted in Fig. 10, the execution of HSeg is stopped at N_{min} regions in the recursive function rhseg(L,X). N_{min} is equal to ¼ the number of pixels in the subimage processed at the deepest level of recursion. In our example N_{min} would equal (1024 × 1024/4 =) 262 144 regions. At the deepest level of recursion (i.e., $L = L_r$), HSeg is normally initialized with each image pixel labeled as a separate region. However, at the other levels of recursion, HSeg is initialized with the segmentation result from the previous step.

Processing window artifact elimination is performed at all but the deepest level of recursion (see Fig. 10). This procedure is described as follows:

1) For each region, identify other regions that may contain pixels that are more similar to it than the region that they are currently in. These regions are placed in a *candidate_region_label* set for each region. This is done by:
   a) scanning the processing window seam between sections processed at the next deeper level of recursion for pixels that are more similar (by a factor of F_{seam}) to the region existing across the processing window seam.
   b) identifying regions that have a dissimilarity between each other less than or equal to $F_{region} \cdot T_{max}$ (T_{max} is the maximum value of the merge threshold, T, encountered so far in HSeg).
2) For each region with a nonempty *candidate_region_label* set, identify pixels in the region that are more similar by a factor of F_{split} to regions in the *candidate_region_label* set than to the region they are currently in. If S_{wght} = 1.0, simply switch the region assignment of these pixels to the more similar region. Otherwise, split these pixels out of their current regions and remerge them through a restricted version of RHSeg (described in the text below).

The default values of F_{seam} = 1.5, F_{region} = 0.0 (no regions selected via this factor) and F_{split} = 1.5 work well for a wide range of images.

In step 2 above, the regions that remain after the dissimilar pixels are split out retain information from the current level of recursion. This information is passed down to the deeper levels of recursion in the operation of the restricted version of RHSeg.

In the restricted version of RHSeg used in step 2, HSeg is initially restricted to merges involving the pixels that were split out from their regions, the regions neighboring these split-out pixels, the region from which the pixel was split out from, and the regions in the *candidate_region_label* set of the

region from which the pixels were split out from. Note that the existing regions that are not exclusively formed from split-out pixels retain global information from the highest level of recursion. HSeg is further restricted by considering pixels along the processing window boundaries to be "contagious," and prohibiting merges between "contagious" pixels (or regions) and other pixels or regions. This "contagious" property is passed on to any pixel or region that attempts to merge with a "contagious" pixel or region. This "contagious pixel" idea was first advanced by Lee [34], [35]. These merges are performed until the largest merge threshold from the previous step is reached, the number of regions becomes less than or equal to $N_{\min}$, or no other merges can be performed (because all remaining regions formed entirely from split-out pixels are "contagious").

If the above stage of the restricted version of HSeg stops before the largest merge threshold from the previous step is reached, or before the number of regions becomes less than or equal to $N_{\min}$, the "contagious" property is set aside and region merging is continued until the largest merge threshold from the previous step is reached, or the number of regions becomes less than or equal to $N_{\min}$.

Finally, if the number of regions equal to $N_{\min}$ is not reached with the above restricted version of HSeg, the unrestricted version of HSeg is performed until the number of regions becomes less than or equal to $N_{\min}$ regions.

The reader may wonder why the "contagious pixel" idea is not used to *prevent* processing window artifacts in the execution of HSeg in RHSeg. This idea was tried, but this approach turned out to be unreliable because oftentimes so many pixels and regions would become "contagious" that the region-growing process would stall before $N_{\min}$ regions could be achieved. This stalling commonly does occur in the restricted version of RHSeg described above. However, there is a safety valve provided in which the merging process may continue after the "contagious" property is set aside.

It should be noted that the processing window artifact elimination step of RHSeg is invoked *after* the HSeg algorithm is performed until the number of regions reaches $N_{\min}$. This reduces the number of pixels that are split out in step 2 above by allowing a number of regions to merge together across the processing window seams. It also can be noted that, with appropriate values for F_{seam}, F_{region} and F_{split}, the inclusion of the processing window artifact elimination step of RHSeg generally no more than doubles the processing time for RHSeg versus a version of RHSeg that does not include this step.

Acknowlegments

The authors would like to thank P. Gamba from the University of Pavia, Pavia, Italy, and D. Landgrebe from Purdue University, West Lafayette, IN, for providing the hyperspectral data.

References

[1] J. Marceau and G. J. Hay, "Remote sensing contributions to the scale issue," *Can. J. of Remote Sens.*, vol. 25, no. 4, pp. 357-366, 1999.

[2] T. Blaschke, S. Lang, and G. J. Hay, Eds., *Object-Based Image Analysis: Spatial Concepts for Knowledge-Driven Remote Sensing Application*. Berlin, Germany: Springer-Verlag, 2008.

[3] J.-M. Beaulieu and M. Goldberg, "Hierarchy in picture segmentation: A stepwise optimal approach," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 11, no. 2, pp. 150-163, Feb. 1989.

[4] J.-M. Beaulieu and M. Goldberg, "Hierarchical picture segmentation by approximation," *Proc. Can. Commun. Energy Conf.*, Montreal, QC, Canada, 1982, pp. 393-396.

[5] J. C. Tilton and S. C. Cox, "Segmentation of remotely sensed data using parallel region growing," *Proc. of IGARSS*, San Francisco, CA, 1983, pp. 130-137.

[6] M. Willebeek-LeMair and A. P. Reeves, "Region growing on a highly parallel mesh-connected SIMD computer," in *Proc. 2nd Symp. Frontiers Massively Parallel Comput.*, Fairfax, VA, 1988, pp. 93-100.

[7] J. C. Tilton, "Image segmentation by iterative parallel region growing with applications to data compression and image analysis," in *Proc. 2nd Symp. Frontiers Massively Parallel Comput.*, Fairfax, VA, 1988, pp. 357-360.

[8] T. Kurita, "An efficient agglomerative clustering algorithm for region growing," *Proc. of MVA, IAPR Workshop on Mach. Vis. Appl.*, Kawasaki, Japan, Dec. 13-15, 1994, pp. 210-213.

[9] J. W. J. Williams, "Heapsort," *Commun. ACM*, vol.7, No. 12, pp. 347-348, Dec. 1964.

[10] G. Meinel and M. Neubert, "A comparison of segmentation programs for high resolution remote sensing data," *Proc. of Commission IV, XXth ISPRS Congr.*, Istanbul, Turkey, Jul. 2004, pp. 1097-1102.

[11] M. Neubert, H. Herold, and G. Meinel, "Evaluation of remote sensing image segmentation quality – Further results and concepts," in *Proc. 1st Int. Conf. OBIA*, Salzburg, Austria, Jul. 2006.

[12] M. Neubert, H. Herold, and G. Meinel, "Assessing image segmentation quality – Concepts, methods and application," in *Object-Based Image Analysis: Spatial Concepts for Knowledge-Driven Remote Sensing Applications*, T. Blaschke, S. Lang, and G. J. Hay, Eds. Berlin, Germany: Springer-Verlag, 2008.

[13] E. Gofman, "Developing an efficient region growing engine for image segmentation," in *Proc.ICIP*, Atlanta, GA, Oct. 2006, pp. 2413-2416.

[14] M. Baatz and A Schape, "Multiresolution segmentation: an optimizing approach for high quality multi-scale segmentation," in *Angewandte Geographich Informationsverarbeitung, XII*, J. Strobl and T. Blaschke, Eds., Heidelberg, Germany: Wichmann, 2000, pp. 12-23.

[15] U. Benz, P. Hofmann, G. Wilhauck, I. Lingenfelder, and M. Heynen, "Multi-resolution, object-oriented fuzzy analysis of remote sensing data for GIS-ready information," *ISPRS J. Photogrammm. Remote Sens.*, vol 58, no. 3/4, pp. 239-258, Jan. 2004.

[16] J. C. Tilton, "Image segmentation by region growing and spectral clustering with a natural convergence criterion," in *Proc. IGARSS*, Seattle, WA, 1998, pp. 1766-1768.

[17] J. C. Tilton, "Parallel implementation of the recursive approximation of an unsupervised hierarchical segmentation algorithm," in *High Performance Computing in Remote Sensing*, A. J. Plaza and C.-I. Chang, Eds., New York: Chapman & Hall, 2007, ch. 5, pp. 97-107.

[18] J. C. Tilton, *RHSeg User's Manual: Including HSWO, HSeg, HSegExtract, HSegReader and HSegViewer*, version 1.55, Jan. 4, 2012, available via email request to James.C.Tilton@nasa.gov.

[19] Y. Tarabalka, J. A. Benediktsson, Jocelyn Chanussot, and J. C. Tilton, "Multiple spectral-spatial classification approach for hyperspectral data," *IEEE Trans. Geosci. Remote Sens.*, vol. 48, no. 11, pp. 4122-4132, Nov. 2010.

[20] G. Camps-Valls and L. Bruzzone, "Kernel-based methods for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 43, no. 6, pp. 1351-1362, June 2005.

[21] C. Chang and C. Lin, LIBSVM–A Library for Support Vector Machine, 2008. [Online]. Available: http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[22] D. A. Landgrebe, *Signal Theory Methods in Multispectral Remote Sensing*, Hoboken, NJ: Wiley, 2003.

[23] Y. Tarabalka, J. A. Benediktsson, and J. Chanussot, "Spectral-spatial classification of hyperspectral imagery based on partitional clustering techniques," *IEEE Trans. on Geosci. Remote Sens.*, vol. 47, no. 8, pp. 2973-2987, Aug. 2009.

[24] S. Tadjudin and D. A. Landgrebe, "Covariance estimation with limited training samples," *IEEE Trans. Geosci.Remote Sens.*, vol. 37, no. 4, pp. 2113-2118, July1999.
[25] V. Vapnik, *Statistical Learning Theory*. New York: Wiley, 1998.
[26] J. A. Richards and X. Jia, *Remote Sensing Digital Image Analysis: An Introduction*, New York: Springer-Verlag, 1999.
[27] J. C. Tilton, D. J. Cook and N. Ketkar, "The integration of graph-based knowledge discovery with image segmentation hierarchies for data analysis, data mining and knowledge discovery, " in *Proc. IGARSS*, Boston, MA, July 2008, pp. III-491 – III-494.
[28] D. Zamalieva, S. Aksoy, and J. C. Tilton, "Finding compound structures in images using image segmentation and graph-based knowledge discovery," *Proc. IGARSS*, Cape Town, South Africa, Jul. 2009, vol. 5, pp V-252 – V-255.
[29] J. Jasiewicz and T.F. Stepinski, "Example-based retrieval of alike land-cover scenes from NLCD2006 database," submitted to *Geosci. Remote Sens. Let.*, 2012.
[30] J. C. Tilton, D. K. Hall, and G. A. Riggs, "Creation of ersatz ground reference data for validating improvements in the MODIS Snow and Ice Product Suite," in *Proc. IGARSS*, Honolulu, HI, July 2010, pp.2371-2374.
[31] J. C. Tilton, D. C. Comer, K. May, and W. Hurst, "Toward automated detection of archaeological sites utilizing remotely sensed imagery," in *Proc. Soc. Amer. Archaeol. 75th Anniv. Meeting*, St. Louis, MO, April 2010.
[32] F. A. Kruse, A. B. Lefkoff, J. W. Boardman, K. B. Heidebrecht, A. T. Shapiro, P. J. Barloon, and A. F. H. Goetz, "The spectral image processing system (SIPS)–Interactive visualization and analysis of imaging spectrometer data, " *Remote Sens. Environ.*, vol. 44, no. 2/3, pp. 145-163, May/Jun. 1993.
[33] J. C. Tilton, *RHSEG User's Manual: Including the core RHSEG open source release, HSEGExtract, HSEGReader and HSEGViewer*, version 1.47, Dec. 2, 2009, available via email request to James.C.Tilton@nasa.gov.
[34] S. Lee, K.-H. Lee, and C. Kim, "Efficient multi-stage system for unsupervised classification and its application of KOMPSAT-I imagery," in *Proc. IGARSS*, Jul. 2000, vol. 5, pp. 2173-2175.
[35] S. Lee, "An unsupervised hierarchical clustering image segmentation and an adaptive image reconstruction system for remote sensing," Ph. D. Thesis, Univ. Texas Austin, Austin, TX, 1990.

Yuliya Tarabalka (S'08-M'10) received the B.S. degree in computer science from Ternopil Ivan Pul'uj State Technical University, Ukraine, in 2005, and M.Sc. degree in signal and image processing from the Grenoble Institute of Technology (INPG), France, in 2007. She received a joint Ph.D. degree in signal and image processing from INPG and in electrical engineering from the University of Iceland, in 2010. From July 2007 to January 2008, she was a Researcher with the Norwegian Defence Research Establishment, Norway. From September 2010 to December 2011, she was a Postdoctoral research fellow with CISTO, NASA Goddard Space Flight Center, Greenbelt, MD, USA.She is currently a Postdoctoral researcher fellow with INRIA Sophia Antipolis - Méditerranée and with the French Space Agency CNES, France. Her research interests are in the areas of image processing, pattern recognition, hyperspectral imaging and development of efficient algorithms.

Paul M. Montesano received a B.A. degree in Geography from Rutgers University in New Jersey in 2001 and an M.S. degree in Environmental Monitoring from the University of Wisconsin-Madison in 2006. He is currently a Ph.D. student in the Department of Geographical Sciences at the University of Maryland-College Park. He worked at the Walton Center for Remote Sensing & Spatial Analysis at Rutgers University from 2001 to 2004 as a research technician on statewide critical habitat mapping projects. Since joining Sigma Space Corporation in 2006, he has worked at the NASA Goddard Space Flight Center, Greenbelt, MD, USA on boreal forest mapping projects, focusing on the remote sensing of forest structure.

James C. Tilton (S'79-M'81-SM'94) received B.A. degrees in electronic engineering, environmental science and engineering, and anthropology and a M. E. E. (electrical engineering) from Rice University, Houston, TX in 1976. He also received an M. S. in optical sciences from the University of Arizona, Tucson, AZ in 1978 and a Ph. D. in electrical engineering from Purdue University, West Lafayette, IN in 1981.

He is currently a Computer Engineer with the Computational and Information Sciences and Technology Office (CISTO) of the Science and Exploration Directorate at the NASA Goddard Space Flight Center, Greenbelt, MD. As a member of CISTO, Dr. Tilton is responsible for designing and developing computer software tools for space and earth science image analysis, and encouraging the use of these computer tools through interactions with space and Earth scientists. His software development has resulted in two patents and two other patent applications.

Dr. Tilton is a senior member of the IEEE Geoscience and Remote Sensing Society (GRSS). From 1992 through 1996, he served as a member of the IEEE GRSS Administrative Committee. Since 1996 he has served as an Associate Editor for the IEEE Transactions on Geoscience and Remote Sensing.

Emanuel Gofman received the M.Sc. degree in applied mathematics from Moscow State University in Russia and the Ph.D. in computer science (technical cybernetics) from Riga Polytechnic Institute in Latvia. After immigrating to Israel in 1977, he joined IBM where he is currently a Research Staff Member with the IBM Haifa Research Labs. Dr. Gofman's work at IBM has focused on developing software tools in diverse areas such as hydraulic networks, verification of hardware designs, lithography, and image processing. His work has been awarded prizes including two from the Information Processing Association in Israel and the prestigious IBM Outstanding Technical Achievement Award. He is the author of several papers, numerous technical reports, and a dozen patents.